Subject: Re: Multidimensional arrays and CALL_EXTERNAL
Posted by Foldy Lajos on Fri, 20 Jun 2008 18:24:59 GMT
View Forum Message <> Reply to Message

On Fri, 20 Jun 2008, Dan wrote:

> Hi everyone,
>
> Now that I have gotten CALL_EXTERNAL to work correctly, I am trying to
> figure out how IDL stores multidimensional arrays in memory. Since
> CALL_EXTERNAL passes an arrays by reference (pointer to the first
> element of the array), the multidimensional arrays in C act like a 1-D
> array. I have figured out how to access elements of a 2D array
> (array(i, j) ---> array[i + i_size * j]), but haven't been able to
> figure out how to access elements from a 4D array.
>
> For example, how would I access element:
> array(i, j, k, m) from the pointer in C?
>

I have never used CALL_EXTERNAL, but my guess is:

```
array(i, j, k, m) ---> array[i
                       + i_size * j
                       + i_size * j_size * k
                       + i_size * j_size * k_size * m]
```

regards,
lajos

Subject: Re: Multidimensional arrays and CALL_EXTERNAL
Posted by Dan[1] on Fri, 20 Jun 2008 19:36:03 GMT
View Forum Message <> Reply to Message

Hi Lajos,

This doesn't seem to be working for me.

Subject: Re: Multidimensional arrays and CALL_EXTERNAL
Posted by Foldy Lajos on Fri, 20 Jun 2008 20:21:32 GMT
View Forum Message <> Reply to Message

On Fri, 20 Jun 2008, Dan wrote:

> Hi Lajos,

>
> This doesn't seem to be working for me.
>

Create a 4D array in IDL: arr=lindgen(2,3,4,5). Pass this array to the C routine, and print the array elements in C. You will see the pattern.

regards,
lajos

Subject: Re: Multidimensional arrays and CALL_EXTERNAL
Posted by Allan Whiteford on Mon, 23 Jun 2008 12:15:34 GMT
View Forum Message <> Reply to Message

Dan wrote:
> Hi Lajos,
>
> This doesn't seem to be working for me.

Dan,

It works for me; as an example:

multi.c

```
#include <stdio.h>
#include "idl_export.h"

void do_work(float *a)
{
        printf("%f\n",a[3+5*10+7*10*20+9*10*20*30]);
}

IDL_VPTR multi(int argc, IDL_VPTR argv[])
{
 do_work((float *) argv[0]);
}
```

multi.pro

```
pro multi
 a=fltarr(10,20,30,40)

 a[3,5,7,9]=3.14
```

```
 junk=call_external('multi.so','multi',a,/unload)
end
```

Compile
---

gcc -I /usr/local/rsi/idl/external/include -shared multi.c -o multi.so

does the above work for you? Can you post an example of code which fails?

Thanks,

Allan
